J. King Saud University, Vol. 22, Comp. & Info. *Sci*., pp. 37-44,Riyadh (1431H./2010)

A Novel Recursive Algorithm For Detail-Preserving Impulse Noise Removal

Naif Alajlan
Advanced Lab for Intelligent Systems Research (ALISR)
College of Computer and Information Sciences, King Saud University
P. O. Box 51178, Riyadh 11543, Saudi Arabia
Email: najlan@ksu.edu.sa

(Received 07/06/2009; accepted for publication 30/12/2009)

***Keywords*:** Impulse noise removal; recursive algorithm; image denoising; nonlinear filtering.

Abstract. Removing impulse noise from digital images without altering image details is a central problem in image processing. There is a huge number of filters in the literature that deal with this problem. In the state-of-the-art filters, the removal of impulse noise is usually carried out in two main steps: detection and estimation. In this paper, we show that we can take any impulse noise removal approach, where the noise detection and estimation processes are separable, and enhance its detail preservation capability. Unlike the classical recursive implementation which performs sequential row-by-row scanning, the proposed algorithm maximizes the contribution of noise-free neighbors in detecting and correcting the noisy pixels. Simulation results using standard test images demonstrate the effectiveness of the proposed algorithm in preserving image details.

1. Introduction

Impulse noise (IN) is a fundamental problem in most image processing applications. There are various sources for IN that occur during the image acquisition, transmission, and storage. Removing IN can be an ultimate goal by itself, for instance, to improve the visual quality of an art work image, or as a preprocessing step for further processing. For example, in many operations that involve computing image derivatives, such as edge detection, any noise in the image can result in serious errors due to noise magnification.

Removing IN is a very challenging task, since IN exhibits a randomness behavior in both pixel's value and spatial location. There is a huge number of techniques that deal with removing IN in digital images, for example, [1–4]. The performance of impulse denoising filters is usually evaluated in terms of noise suppression or elimination, details preservation, true-value estimation, visual quality, and computational complexity. The choice of the proper filter for a given application is a tradeoff between these performance indices. In the state-of-the-art filters, the removal of IN is usually carried out in two main stages. The first is detection that examines the presence of noise. Then, if the pixel at hand is noisy, a correction stage estimates the original value of that pixel.

Unlike the classical median filter [5], the detection stage avoids altering noise-free pixels which results in preserving the image details. As in most local image processing operations, the filtering of IN is performed by scanning the image in a sequential row-by-row basis. In the recursive implementation, half of the neighborhood pixels are already filtered and the other half is still unprocessed, regardless of their noise status. Therefore, the presence of noisy pixel(s) among the unprocessed portion can alter the detection and correction processes.

In this paper, a new recursion approach for removing IN is proposed, which works for any IN removal approach with distinct detection and

correction processes. The main aim is to maximize the contribution of noise-free neighbors in detecting and correcting noisy pixels. Intuitively, image regions with lower noise density are filtered before regions with higher noise density. The rest of the paper is organized as follows. Section 2 explains the proposed algorithm in details. Then, Section 3 demonstrates the performance of the proposed algorithms via experimental results. Finally, Section 4 concludes the work of this paper.

2. The Proposed Algorithm

Given an arbitrary impulse noise removal filter with separable noise detection and correction operations, denoted by *NoiseMap* and *Estimate*, respectively. Our aim is to enhance the filter's performance by applying the proposed recursive algorithm, which includes two main steps. In the first, an initial set of noisy pixels is identified, which is called the noise map[1]. Then, these noisy pixels are recursively corrected based on the number of noise-free neighbors. Since the initial noise detection is non-recursive, the noisy neighbors of each corrected noisy pixel are re-examined to verify whether they are still noisy or not.

Algorithm 1 shows a pseudo code of the proposed algorithm. It accepts, as input, an image corrupted with impulse noise, F . Then, an initial set of noisy pixels are identified using the function *NoiseMap*, which returns the noise map of the corrupted image, M . To sort the noisy pixels according to the number of their noise-free neighbors, the locations of the noisy pixels are listed in the set, B. An auxiliary set, E, is attached to B such that each element $\mathrm{E}_k \in \mathrm{E}$ stores the locations of the noisy neighbors to B_k, if exist. The main algorithm recursively corrects the noisy pixels in B as follows. At first, a noisy pixel $\mathrm{B}_p \in \mathrm{B}$ with maximum number of noise-free neighbors, i.e., the cardinality of E_p is minimum, is selected and its original value is estimated using the function *Estimate*. Then, B_p is removed from B and the list E_i that belongs to the *i*-th member in E_p, if any. Since the algorithm is recursive, the noise status of the corrected pixel's noisy neighbors is updated using the recursive algorithm *fUpdate*, which is shown in Algorithm 2. If the noise status of any noisy neighbor to the given pixel changes, i.e., becomes noise-free, the neighbor pixel is removed from B and E as done for the corrected pixel. Then, the algorithm executes itself again for the noisy neighbors of the updated pixel. Note that the function *fUpdate* does not estimate noisy pixels; therefore, the recursion of the function only works when the employed noise detection approach excludes the (processed) noisy neighbors from the detection process. In this case, the change of noise status of a noisy pixel could have an impact on the noise status of the noisy neighbors. The main algorithm iterates until all noisy pixels are estimated, i.e., $\mathrm{B} = \varnothing$ becomes empty.

[1]The noise map of a noisy image is a binary image with the same size as the noisy image where foreground and background pixels represent noisy and noise-free pixels, respectively.

3. The Modified Peak and Valley Filter

In this paper, the modified peak and valley filter (*MPVF*) [6] is considered as case study since this filter does not require any parameter tuning and to avoid any bias related to parameter selection when applying the proposed algorithm. The *MPVF* performs a noise detection test on each pixel following a row-by-row scanning; then, if the pixel is noisy, its original value is estimated using a minimum-maximum approach. It should be noted that we are not interested in the effectiveness of the *MPVF*, but rather, how the proposed algorithm improves the performance of the *MPVF*. The impulse noise is of the salt-and-pepper type, i.e., noisy pixels take one of the two extreme values with equal probability. In the remainder of this section, a brief description about the detection and estimation processes of the *MPVF* is given.

The MPV F is a non-linear, non-iterative filter that is based on order statistics to remove impulsive noise from an image. The filter operates in two steps. In the first step, a pixel is considered noisy if its gray level value is either greater (peak) or smaller (valley) than the gray levels of its neighbors. Then, the corrupted pixel's gray level value is estimated using a recursive minimum maximum method. This noise detection approach is parameter-free and non-iterative which enables the

filter to be applicable to all images. In the second step, the recursive minimum maximum method provides an estimate of the corrupted pixels at constant signal as well as edges even when the noise probability is high. More specifically, the *MPVF* for impulsive noise filtering works as follows:

1. For a 3 × 3 window centered at the test pixel, as shown in Fig. 1.

2. If $d_9 \geq \max(d_i)$ or $d_9 \leq \min(d_i)$, where $1 \leq i \leq 8$, then d_9 is a noisy pixel and must be estimated, go to step 3. Otherwise $y = d_9$.

3. When a noisy pixel is detected, its gray level is estimated as follows. For $1 \leq i \leq 4$, let $L_i = \max(d_i, d_{9-i})$ and $E_i = \min(d_i, d_{9-i})$. Set $P_{\min} = \min(L_1, .., L_4)$ and $P_{\max} = \max(E_1, .., E_4)$. Then $y = (P_{\max} + P_{\min})/2$.

Note that if there are three identical noisy pixels along one direction within the window, then the output of the filter is largely influenced by the noisy pixels. In this case, either $P_{\max}$ or $P_{\min}$ is equal to the level of the noisy pixel. In the classical recursive implementation, d_1, d_2, d_3, and d_4 in Fig. 1 are the previous outputs of the filter and d_5, d_6, d_7, and d_8 are the original unprocessed image data. The latter group can undermine both the detection and estimation processes if it contains noisy pixels. Alternatively, the proposed algorithm overcomes this defect since if the test pixel d_9 has a noisy neighboring pixel d_i, where $1 \leq i \leq 8$, then the latter must be a pervious output of the filter provided that it has a smaller number of noisy neighboring pixels than that of d_9.

Algorithm 1 Labeling noisy pixels:
$\mathrm{G} = fCorrect(\mathrm{F})$

Notation:

F is an input noisy image to be filtered.
G is the output image.
$\mathrm{B} \equiv \{\mathrm{B}_k\}$ is a list of the locations of noisy pixels.
E is a set attached to B where E_k is the set of noisy neighbors to B_k.
$\mathrm{N}(r,c)$ is the neighborhood of the location (r,c).
NoiseMap is an arbitrary function that returns the noise map of F.
Estimate is an arbitrary function that estimates the original value of a corrupted pixel.
$|\mathrm{E}_k|$ is the cardinality of E_k.

$\mathrm{B} \leftarrow \varnothing$
$\mathrm{E} \leftarrow \varnothing$
$\mathrm{G} \leftarrow \mathrm{F}$
$\mathrm{M} \leftarrow NoiseMap(\mathrm{F})$
$k \leftarrow 0$
for all $\mathrm{M}(r,c) = 1$ **do**
  $k \leftarrow k + 1$
  $\mathrm{B}_k \leftarrow (r,c)$
  for all $\mathrm{M}(i,j) \in \mathrm{N}(r,c)$ **do**
    if $\mathrm{M}(i,j) = 1$ **then**
      $\mathrm{E}_k \leftarrow \mathrm{E}_k \bigcup \{(i,j)\}$
    end if
  end for
end for
while $\mathrm{B} \neq \varnothing$ **do**
  $p \leftarrow k$ with $|\mathrm{E}_k|$ is minimum
  $(r_p, c_p) \leftarrow \mathrm{B}_p$
  $m_p \leftarrow \mathrm{E}_p$
  $\mathrm{G}(r_p, c_p) \leftarrow Estimate(\mathrm{G}(r_p, c_p))$
  $\mathrm{B} \leftarrow \mathrm{B} - \{\mathrm{B}_p\}$
  $\mathrm{E} \leftarrow \mathrm{E} - \{\mathrm{E}_p\}$
  if $|m_p| \neq 0$ **then**
    for all $(r_i, c_i) \in m_p$ **do**
      $\mathrm{E}_i \leftarrow \mathrm{E}_i - \{(r_p, c_p)\}$
    end for
    $[\mathrm{B}, \mathrm{E}] = fUpdate(\mathrm{B}, \mathrm{E}, m_p)$
  end if
end while
return G

Algorithm 2 Updating the noise list:
$[\mathrm{B}, \mathrm{E}] = fUpdate(\mathrm{B}, \mathrm{E}, m_p)$

1: **for** all $(r_i, c_i) \in m_p$ **do**
2: **if** B_i is noise-free **then**
3: $m_q \leftarrow E_i$
4: $B \leftarrow B - \{B_i\}$
5: $E \leftarrow E - \{E_i\}$
6: **if** $|m_p| \neq 0$ **then**
7: **for** all $(r_j, c_j) \in m_q$ **do**
8: $E_j \leftarrow E_j - \{(r_j, c_j)\}$
9: **end for**
10: $[B, E] = fUpdate(B, E, m_q)$
11: **end if**
12: **end if**
13: **end for**
14: **return** B, E

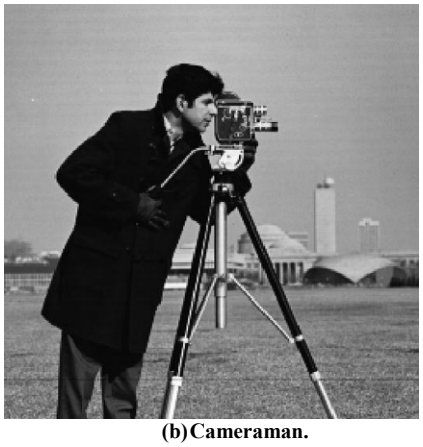
d_1	d_2	d_3
d_4	d_9	d_5
d_6	d_7	d_8

Fig. 1. Window used to detect and process impulsive noisy pixels.

1. Experimental Results

In this section, the performance of the proposed algorithm is tested on the modified peak and valley filter (*MPVF*) [6] as case study. Let $MPVF$ denotes the implemented *MPVF* using the proposed algorithm. We tested the performance of both *MPVF* and $MPVF$ on three standard images used by the image processing research community as shown in Fig. 2. The first image is the first frame of a public domain twelve-frame sequence, known as *Hamburg Taxi* (190 × 256 pixels), the second is the *Cameraman* image (256 × 256 pixels), and the third is the well-known *Lena* image (512 × 512 pixels). These images contain a nice mixture of detail, flat regions, shading, and texture that do a good job of testing various image processing algorithms. We restricted our tests to a 3 × 3 window size. Fig. 3 shows outcomes of the $MPVF$ and *MPVF* applied to the *Cameraman* image at impulsive noise probability of 60 %.

(a) Hamburg Taxi.

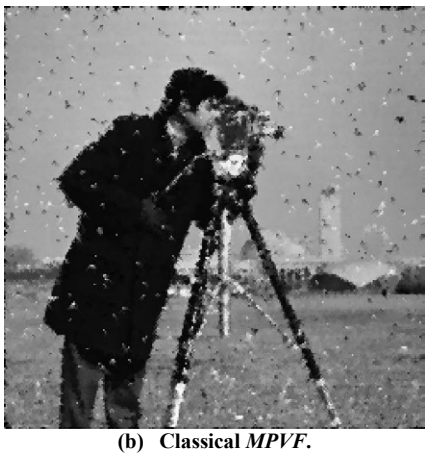


(b) Cameraman.

(c) Lena.

Fig. 2. Test images.

(a) 60% corrupted.



(b) Classical *MPVF*.

(c) Proposed $MPVF$.

Fig. 3. Filtering results of the Cameraman image.

To assess the quality of the visual appearance, four performance measures are used to compare the filters [7]: the percentage of noisy pixels replaced by the true values (n_t), the percentage of noisy pixels attenuated (n_a), the percentage of true pixels modified (t_m), and the mean squared error between the original noise-free and filtered images (*MSE*). All images were corrupted with impulsive noise with probability ranging from 1 % to 90 %. Figs. 4, 5, and 6 show the plots of the four performance measures versus the impulsive noise probability for the *Hamburg Taxi*, *Cameraman*, and *Lena* images, respectively. In terms of noise attenuation, the proposed and classical algorithms have comparable performance. This is due to the fact that the classical approach achieves near 100% noise attenuation. The noise elimination is also comparable since both filters employ the same noise estimator. As shown in all figures, the proposed algorithm achieves significant improvement in terms of details preservation with less modified noise-free pixels by a considerable margin. Finally, the proposed algorithm outperforms the classical algorithm in the *MSE* sense.

In terms of computation time, Table 1 shows that the proposed algorithm demands more computations than the classical algorithm. The algorithms were implemented in MATLAB 7.4 on PC workstation with a Pentium Duo 2GHz and 2GB RAM, running Windows Vista. It should be noted that these times are for demonstration only since the codes were not optimized and better speeds can be obtained using other programming languages such as C and C++.

Table 1. Execution times (in seconds), averaged over 1% to 90% noise levels, of the classical and proposed algorithms applied on different images.

Algorithm	Hamburg Taxi	Camera-man	Lena
Classical	1.9	2.5	8.2
Proposed	3.6	4.6	13.9

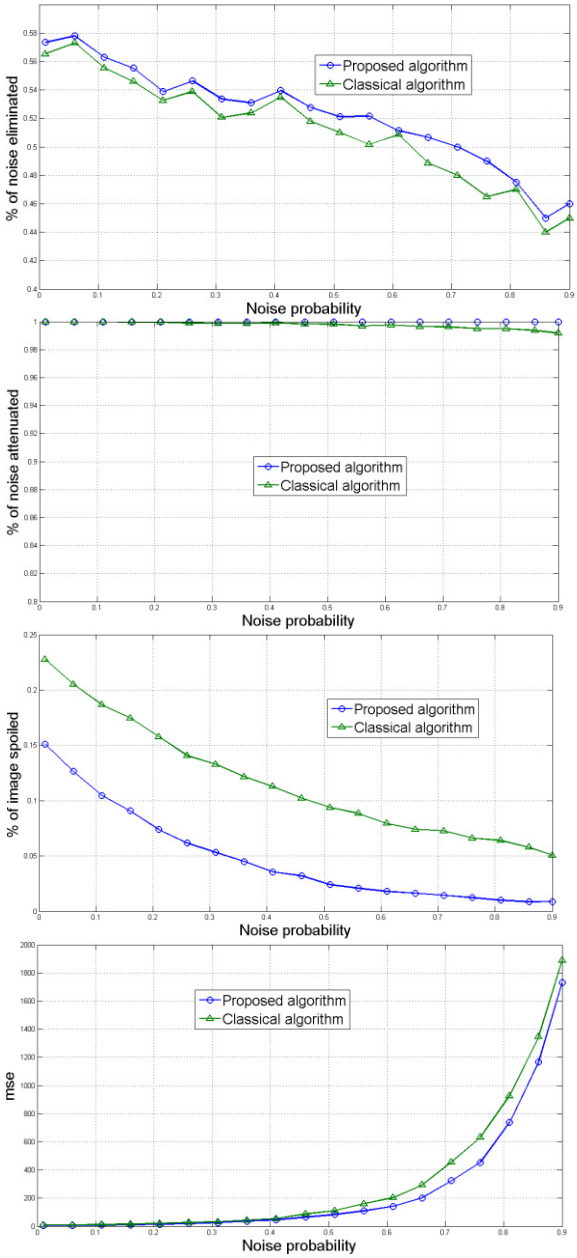
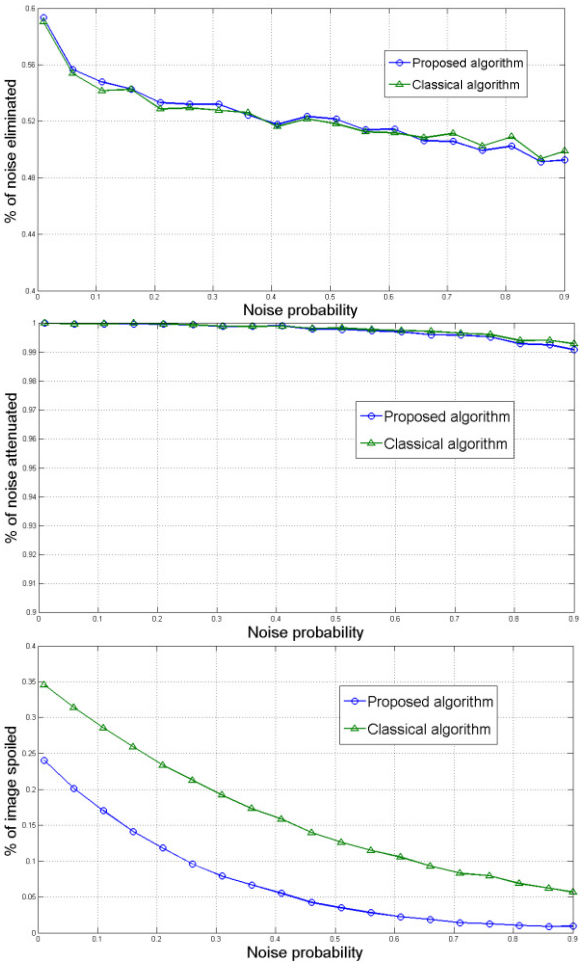


Fig. 4. Performance of the proposed algorithm on the *Hamburg Taxi* image.

2. Conclusions

In this work, a new recursion algorithm for removing impulse noise in digital images is proposed. The algorithm gives priority in the estimation to the noisy pixels with greater number of noise-free neighbors; therefore, the contribution of noise-free neighbors in detecting and correcting the noisy pixels is maximized. Simulation results demonstrated the effectiveness of the proposed algorithm in preserving image details. However, the proposed algorithm demands more computations. Our future work in this area includes testing the algorithm with random noise and using other state-of-the-art impulse noise removal filters.

Acknowledgement: The author would like to acknowledge the support of the Research Center of the College of Computer and Information Sciences at King Saud University.



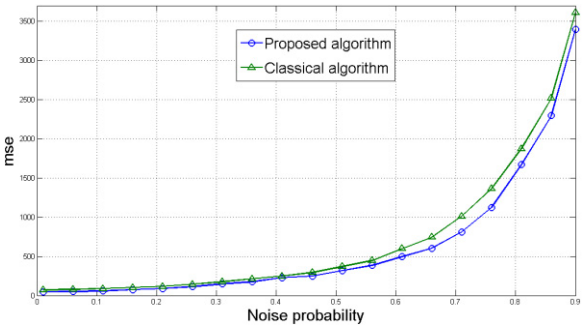
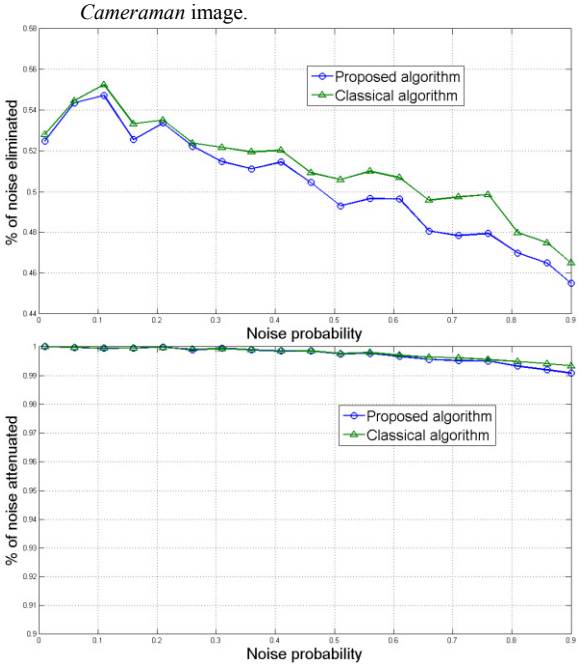


Fig. 5. Performance of the proposed algorithm on the *Cameraman* image.



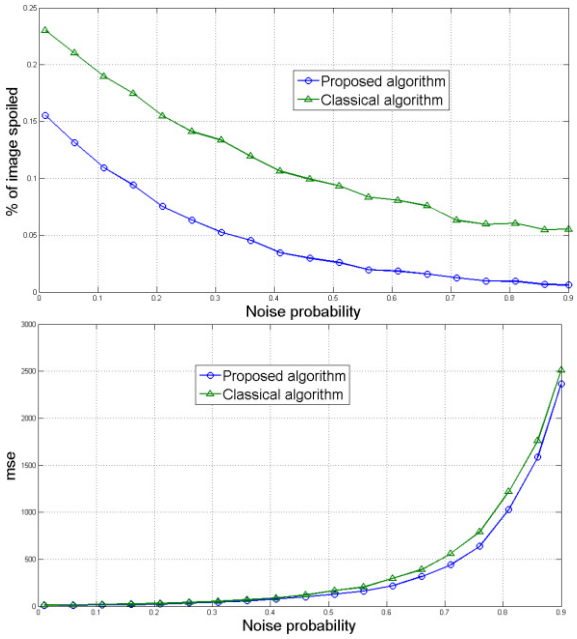


Fig. 6. Performance of the proposed algorithm on *Lena* image.

References

[1] Xu, Y. and Lae, E.M. (1998). "Restoration of images contaminated by mixed gaussian and impulse noise using a recursive minimum-maximum method". IEEE Proc. Vision, Image and Signal Processing, 145(4): 264–270.

[2] Moreno, H.G., Bascon, S.M., Manso, M.U., and Martin, P.M. (2001). "Elimination of impulsive noise in images by means of the use of support vector machines". XVI National Symposium of URSI, (1):1-2.

[3] Andreadis, I. and Louverdis, G. (2004). "Real-time adaptive image impulse noise suppression". IEEE Transaction on Instrumentation and Measurement 53(3): 798 – 806.

[4] Luo, W. (2007). "An efficient algorithm for the removal of impulse noise from corrupted images". International Journal of Electronics and Communication, (61): 551–555.

[5] Bovik, A.C., Huang, T., and Munson, D. (1983). "A generalization of median filtering using linear combinations of order statistics". IEEE Transactions on Acoustics, Speech, and Signal Processing, 31(6): 1342–1350.

[6] Alajlan, N., Kamel, M.S., and Jernigan, E. (2004). "Detail preserving impulse noise removal". Signal Processing: Image Communication, 19(10): 993–1003.

[7] Windyga, P.S. (2001). "Fast impulsive noise removal". IEEE Transaction on Image Processing, 10(1): 173–179.

خوارزمية تكرارية جديدة للحصول على إزالة الضجيج النبضي مع الحفاظ على التفاصيل

نايف بن عبدالرحمن العجلان

قسم هندسة الحاسب، كلية علوم الحاسب والمعلومات

جامعة الملك سعود، ص.ب: ٥١١٧٨، الرياض ١١٥٤٣، المملكة العربية السعودية

najlan@ksu.edu.sa

(قدم للنشر في ٢٠٠٩/٦/٧م؛ وقبل للنشر في ٢٠٠٩/١٢/٣٠م)

ملخص البحث. تعتبر إزالة الضجيج النبضي من الصور الرقمية دون تغيير تفاصيل الصورة مشكلة مركزية في معالجة الصور. يوجد عدد كبير من المرشحات المنشورة التي تتعامل مع هذه المشكلة. تتم عادة إزالة الضجيج النبضي في المرشحات الحديثة في خطوتين رئيسيتين: كشف وتقدير. نبرهن في هذا البحث أن بإمكاننا أن نأخذ أي طريقة لإزالة الضجيج النبضي، حيث يمكن الفصل بين عمليتي كشف وتقدير الضجيج، ونعزز قدرتها على حفظ التفاصيل. خلافا للتنفيذ التكراري التقليدي والذي يؤدي مسح تسلسلي لكل صف على حده، تقوم الخوارزمية المقترحة بتعظيم مساهمة الجوار الخالي من الضجيج في كشف وتصحيح النقاط ذات الضجيج. تبرهن نتائج المحاكاة باستخدام صور اختبار معيارية فعالية الخوارزمية المقترحة في الحفاظ على تفاصيل الصورة.